Claims

Claims 12-14, and 16-22 have been allowed. Claim 13 has been amended. All allowed/pending claims are reproduced below.

1-11. (Canceled)

12. (Previously Presented) The computer implemented method of claim 22, wherein inferring, prior to run-time, the array shape-tuple of the result of the program expression by creating the shape-tuple expression comprising the input shape-tuple of said each operand and the shape-tuple operator comprises the steps of:

determining a rank of the resulting shape-tuple; and,

promoting the input shape-tuple of said each operand to an appropriate rank.

13. (Currently Amended) The computer implemented method of claim ~~22~~ 12, wherein determining the rank of the resulting shape-tuple comprises the steps of:

determining a rank of the input shape-tuple of said each operand;

identifying an operator corresponding to said each operand; and

determining the rank of the shape-tuple of the result of the program expression according to the operator and the rank of the input shape-tuple of said each operand.

14. (Previously Presented) The computer-implemented method of claim 12, wherein promoting the input shape-tuple of said each operand to the appropriate rank comprises the step of:

comparing the rank of the shape-tuple of the result of the program expression to the rank of the input shape-tuple of said each operand;

responsive to the rank of the shape-tuple of the result of the program expression being greater than the rank of the input shape-tuple of said each operand, expanding the input shape-tuple of said each operand to correspond with the rank of the shape-tuple of the result of the program expression; and, appending trailing extents of the expanded input shape-tuple of said each operand with an appropriate value.

15. (Canceled)

16. (Previously Presented) The computer-implemented method of claim 22, wherein the step of mapping the program operator to the associated shape-tuple operator comprises:

looking up, in a table, the shape-tuple operator corresponding to the program operator.

17. (Previously Presented) The computer-implemented method of claim 22, further comprising the step of calculating a shape predicate for the shape-tuple of the result of the program expression.

18. (Previously Presented) The computer-implemented method of claim 22, further comprising the steps of:

performing an array conformability check at run-time for a first program expression; and

applying a result of the array conformability check to a second program expression.

20. (Previously Presented) The computer-implemented method of claim 22, further comprising the step of:

preallocating storage for said each operand whose size is statically unknown, based upon the input shape-tuple of said each operand in a loop.

21. (Previously Presented) The computer-implemented method of claim 14, further comprising:

responsive to the rank of the shape-tuple of the result of the program expression being less than the rank of the input shape-tuple of said each operand, truncating the input shape-tuple of said each operand corresponding with the rank of the shape-tuple of the result of the program expression.

22. (Previously Presented) A computer-implemented method for inferring, prior to run-time, an array shape of a result of a program expression of a high-level array-based language, the method comprising:

arranging an extent for each array dimension of each operand of the program expression of the high-level array-based language when the size of at least one of said each operand is unknown into an input shape-tuple of said each operand;

identifying a program operator associated with said each operand in the program expression;

mapping the program operator to an associated shape-tuple operator, wherein the shape-tuple operator is based upon the shape semantics of the program operator; and,

inferring, prior to run-time, an array shape-tuple of the result of the program expression by creating a shape-tuple expression comprising the input shape-tuple of said each operand and the shape-tuple operator.